

Phân loại thư rác sử dụng thuật toán SVM

Nhóm 3

B22DCKH024 - Vũ Công Tuấn Dương
B22DCCN768 - Nguyễn Sơn Tùng
B22DCCN479 - Nguyễn Đức Lâm
B22DCCN347 - Trần Đức Hoàng
B22DCCN348 - Trần Huy Hoàng

Ngày 11 tháng 5 năm 2025

- ① Giới thiệu
- ② Lý thuyết
- ③ Cài đặt
- ④ Triển khai và demo

- 1 Giới thiệu
- 2 Lý thuyết
- 3 Cài đặt
- 4 Triển khai và demo

Dữ liệu được lấy trên Kaggle

- Combined Spam Email CSV of 2007 TREC Public Spam Corpus and Enron-Spam Dataset
- 83448 bản ghi email bằng tiếng Anh được phân loại thành 2 nhãn là Spam và Non-spam trong đó số email spam: 43910 và số email ham:(không spam) là 39538

Mục tiêu

- Xây dựng được mô hình Linear SVM cơ bản để phân loại
- Đánh giá mô hình dựa trên các thang đo như độ chính xác và F1 score
- Demo được trên giao diện web

Với một bộ dữ liệu đã chuẩn hóa hoặc chuẩn hóa, các siêu phẳng này có thể được mô tả bằng các phương trình:

-

(mọi điểm trên hoặc phía trên ranh giới này thuộc về một lớp với nhãn 1)

-

(mọi điểm trên hoặc phía dưới ranh giới này thuộc về lớp còn lại, với nhãn -1).

Về mặt hình học, khoảng cách giữa 2 siêu phẳng này là $\frac{2}{\|w\|}$ nên ta cần tối thiểu hóa $\|w\|$.

Ràng buộc hàm mục tiêu

Cũng cần ngăn không cho các điểm dữ liệu rơi vào margin, vì vậy phải thêm vào ràng buộc sau: với mỗi i , nếu $y_i = 1$, thì phải có:

$$w^T x_i + b \geq 1$$

hoặc nếu $y_i = -1$, thì phải có:

$$w^T x_i + b \leq -1$$

Ràng buộc này yêu cầu mỗi điểm dữ liệu phải nằm ở phía đúng của margin. Hay có thể viết lại là:

$$y_i(w^T x_i + b) \geq 1, \quad \forall 1 \leq i \leq n$$

Bài toán tối ưu

Tóm lại, ta cần giải bài toán tối ưu:

$$\min_{w,b} \frac{1}{2} \|w\|^2$$

với ràng buộc:

$$y_i(w^T x_i + b) \geq 1 \quad \forall i \in \{1, \dots, n\}$$

Cần tìm w và b

Hàm Lagrange

$$\mathcal{L}(w, b, \lambda) = \frac{1}{2} \|w\|^2 - \sum_{i=1}^N \lambda_i (y_i (w \cdot x_i + b) - 1)$$

$\lambda_i \geq 0$ là các hệ số Lagrange

Điều kiện KKT

$$\lambda_i \geq 0 \quad \forall i = 1, \dots, n$$

$$y^{(i)}(\mathbf{w} \cdot \mathbf{x}^{(i)} + b) - 1 \geq 0 \quad \forall i = 1, \dots, n$$

$$\lambda_i [y^{(i)}(\mathbf{w} \cdot \mathbf{x}^{(i)} + b) - 1] = 0 \quad \forall i = 1, \dots, n$$

Phương trình Lagrange và điều kiện KKT

- $\forall i : \lambda_i = 0$ hoặc $y^{(i)}(\mathbf{w} \cdot \mathbf{x}^{(i)} + b) = 1$.
- Các điểm sao cho $\lambda_i > 0$ nằm trên margin được gọi là các vector hỗ trợ (support vectors)

Điều kiện KKT

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^n \lambda_i y^{(i)} \mathbf{x}^{(i)} = \mathbf{0} \quad (1)$$

$$\frac{\partial \mathcal{L}}{\partial b} = - \sum_{i=1}^n \lambda_i y^{(i)} = 0 \quad (2)$$

Từ (1):

$$\mathbf{w} = \sum_{i=1}^n \lambda_i y^{(i)} \mathbf{x}^{(i)} \quad (3)$$

Từ (2):

$$\sum_{i=1}^n \lambda_i y^{(i)} = 0 \quad (4)$$

Điều kiện KKT

$$\begin{aligned} L(w, b, \lambda_1, \dots, \lambda_n) &\equiv \frac{1}{2} \|w\|^2 - \sum_{i=1}^n \lambda_i \{y^{(i)}(w^\top x^{(i)} + b) - 1\} \\ &= \frac{1}{2} \|w\|^2 - \underbrace{\sum_{i=1}^n \lambda_i y^{(i)} x^{(i)\top}}_{=w^\top} w - \underbrace{\sum_{i=1}^n \lambda_i y^{(i)} b}_{=0} + \sum_{i=1}^n \lambda_i \end{aligned}$$

Điều kiện KKT

$$\begin{aligned} L(w, b, \lambda_1, \dots, \lambda_n) &\equiv \frac{1}{2} \|w\|^2 - \|w\|^2 + \sum_{i=1}^n \lambda_i \\ &= \sum_{i=1}^n \lambda_i - \frac{1}{2} \|w\|^2 \\ &= \sum_{i=1}^n \lambda_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \lambda_i \lambda_j y^{(i)} y^{(j)} x^{(i)\top} x^{(j)} \\ &\equiv \tilde{L}(\lambda_1, \dots, \lambda_n) \end{aligned}$$

Quy về bài toán

Có thể quy về bài toán tìm:

$$\begin{aligned}\hat{\lambda} &= \arg \max_{\lambda} \tilde{L}(\lambda) \\ &= \arg \max_{\lambda} \left\{ \sum_{i=1}^n \lambda_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \lambda_i \lambda_j y^{(i)} y^{(j)} \mathbf{x}^{(i)T} \mathbf{x}^{(j)} \right\}\end{aligned}$$

với điều kiện $\lambda_i \geq 0, \sum_{i=1}^n \lambda_i y^{(i)} = 0, (i = 1, 2, \dots, n)$.

Ước lượng λ bằng phương pháp Gradient Descent

Nguyên lý phương pháp

- Phương pháp Gradient Descent được sử dụng để tìm nghiệm tối ưu λ
- Các giá trị tham số ban đầu $\lambda^{[0]}$ được đặt ngẫu nhiên
- Cập nhật theo hướng gradient (vì đây là bài toán tối đa hóa):

$$\lambda^{[t+1]} = \lambda^{[t]} + \eta \tilde{L}(\lambda)$$

- η là tốc độ học (learning rate)

Biểu diễn vector cho bài toán SVM

Ma trận dữ liệu và vector

$$\mathbf{X}_{[n \times p]} = \begin{pmatrix} x_1^{(1)} & x_2^{(1)} & \cdots & x_p^{(1)} \\ x_1^{(2)} & x_2^{(2)} & \cdots & x_p^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{(n)} & x_2^{(n)} & \cdots & x_p^{(n)} \end{pmatrix} = \begin{pmatrix} -\mathbf{x}^{(1)T} - \\ -\mathbf{x}^{(2)T} - \\ \vdots \\ -\mathbf{x}^{(n)T} - \end{pmatrix}$$
$$\mathbf{y}_{[n \times 1]} = \begin{pmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(n)} \end{pmatrix}, \quad \lambda_{[n \times 1]} = \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_n \end{pmatrix}$$

Ma trận H và phép nhân Hadamard

Định nghĩa ma trận H

$$\mathbf{H}_{[n \times n]} \equiv \mathbf{y}_{[n \times 1]} \mathbf{y}_{[1 \times n]}^T \odot \mathbf{X}_{[n \times p]} \mathbf{X}_{[p \times n]}^T$$

- \odot là tích Hadamard (phép nhân từng phần tử)
- Các phần tử của ma trận: $(H)_{ij} = y^{(i)} y^{(j)} \mathbf{x}^{(i)T} \mathbf{x}^{(j)}$
- H là ma trận đối xứng

Viết lại hàm Lagrangian

Biểu diễn hàm Lagrangian dưới dạng vector

$$\begin{aligned}\tilde{L}(\lambda) &\equiv \sum_{i=1}^n \lambda_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \lambda_i \lambda_j y^{(i)} y^{(j)} \mathbf{x}^{(i)T} \mathbf{x}^{(j)} \\ &= \sum_{i=1}^n \lambda_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \lambda_i \lambda_j (H)_{ij} \\ &= \|\lambda\| - \frac{1}{2} \lambda^T H \lambda\end{aligned}$$

Với $\|\lambda\| = \sum_{i=1}^n \lambda_i$ là tổng các phần tử của vector λ

Tính toán vector gradient

Đạo hàm của hàm Lagrangian theo λ

$$\begin{aligned}\tilde{L}(\lambda)\lambda &= \lambda\|\lambda\| - \frac{1}{2}\lambda\lambda^T H\lambda \\ &= \mathbf{1} - H\lambda\end{aligned}$$

Trong đó $\mathbf{1}$ là vector cột với tất cả các thành phần bằng 1: $\mathbf{1} = (1, 1, \dots, 1)^T$

Quy tắc cập nhật trong Gradient Descent

Quy tắc cập nhật cho nhân tử λ

$$\lambda^{[t+1]} = \lambda^{[t]} + \eta(\mathbf{1} - H\lambda^{[t]})$$

- Cập nhật này được lặp đi lặp lại cho đến khi hội tụ
- Cần đảm bảo ràng buộc $\lambda_i \geq 0$ bằng cách cắt giá trị âm về 0

Tìm $\hat{\mathbf{w}}$ từ $\hat{\lambda}$

Tính vector trọng số $\hat{\mathbf{w}}$

Từ điều kiện KKT, ta có:

$$\hat{\mathbf{w}} = \sum_{i=1}^n \hat{\lambda}_i y^{(i)} \mathbf{x}^{(i)}$$

- Từ điều kiện KKT (3):

$$\hat{\lambda}_i = 0, \text{ hoặc } y^{(i)}(\mathbf{w}^T \mathbf{x}^{(i)} + b) - 1 = 0$$

- Dữ liệu $\mathbf{x}^{(i)}$ được phân loại:

$$\begin{cases} \hat{\lambda}_i \neq 0 \Leftrightarrow \mathbf{x}^{(i)} \text{ là vector hỗ trợ,} \\ \hat{\lambda}_i = 0 \Leftrightarrow \mathbf{x}^{(i)} \text{ không phải là vector hỗ trợ.} \end{cases}$$

Tìm $\hat{\mathbf{w}}$ từ $\hat{\lambda}$

Tính vector trọng số $\hat{\mathbf{w}}$

Từ điều kiện KKT, ta có:

$$\hat{\mathbf{w}} = \sum_{i=1}^n \hat{\lambda}_i y^{(i)} \mathbf{x}^{(i)}$$

- Chỉ tính tổng trên các vector hỗ trợ:

$$\hat{\mathbf{w}} = \sum_{\mathbf{x}^{(i)} \in S} \hat{\lambda}_i y^{(i)} \mathbf{x}^{(i)}$$

Tìm \hat{b} từ các vector hỗ trợ

Tính tham số \hat{b}

$$\begin{aligned}\hat{b} &= \frac{1}{y^{(i)}} - \hat{\mathbf{w}}^T \mathbf{x}^{(i)} \\ &= y^{(i)} - \hat{\mathbf{w}}^T \mathbf{x}^{(i)} \quad (\text{vì } y^{(i)} = 1 \text{ hoặc } -1)\end{aligned}$$

- Trong thực tế, để giảm sai số, tính trung bình trên tất cả các vector hỗ trợ:

$$\hat{b} = \frac{1}{|S|} \sum_{\mathbf{x}^{(i)} \in S} (y^{(i)} - \hat{\mathbf{w}}^T \mathbf{x}^{(i)})$$

- $|S|$ là số lượng vector hỗ trợ

Tóm tắt thuật toán Gradient Descent cho SVM

- ➊ **Khởi tạo:** $\lambda^{[0]}$ ngẫu nhiên, tính ma trận H
- ➋ **Lặp:** Cập nhật λ theo công thức:

$$\lambda^{[t+1]} = \lambda^{[t]} + \eta(\mathbf{1} - H\lambda^{[t]})$$

- ➌ **Áp dụng ràng buộc:** $\lambda_i \geq 0$
- ➍ **Tìm vector hỗ trợ:** $S = \{i : \lambda_i > 0\}$
- ➎ **Tính tham số mô hình:**

$$\hat{\mathbf{w}} = \sum_{\mathbf{x}^{(i)} \in S} \hat{\lambda}_i y^{(i)} \mathbf{x}^{(i)}$$
$$\hat{b} = \frac{1}{|S|} \sum_{\mathbf{x}^{(i)} \in S} (y^{(i)} - \hat{\mathbf{w}}^T \mathbf{x}^{(i)})$$

① Giới thiệu

② Lý thuyết

Lý thuyết SVM

Thuật toán Pegasos

Thuật toán Pegasos cơ bản

Bước chiếu

Hàm mục tiêu và hàm mất mát Hinge

③ Cài đặt

④ Triển khai và demo

Mô tả bằng mã giả

Algorithm 1 The Pegasos algorithm.

Inputs: a list of example feature vectors X
a list of outputs Y
regularization parameter λ
the number of steps T

$w = (0, \dots, 0)$
for t in $[1, \dots, T]$
 select a position i randomly
 $\eta = \frac{1}{\lambda \cdot t}$
 score = $y_i \cdot (w \cdot x_i)$
 if score < 1
 $w = (1 - \eta \cdot \lambda) \cdot w + (\eta \cdot y_i) \cdot x_i$
 else
 $w = (1 - \eta \cdot \lambda) \cdot w$
the end result is w

Hình 2: Mô tả thuật toán Pegasos cơ bản bằng mã giả

③ Cài đặt

Tiền xử lý và chuẩn bị dữ liệu

Load dữ liệu

Tiền xử lý dữ liệu

Chuẩn bị dữ liệu

Mô hình HardMarginSVM

Mô hình SVM tối ưu bởi thuật toán Pegasos

Chạy với bộ dữ liệu ban đầu và lưu mô hình

④ Triển khai và demo

Import thư viện

In [1]:

```
import pandas as pd
import nltk
nltk.download('stopwords')
nltk.download('punkt_tab')
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
import re
from nltk.stem import SnowballStemmer

#Visualization
import matplotlib.pyplot as plt

#Feature Engineering
import string
import re
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split

#Evaluation Metric
from sklearn.metrics import accuracy_score, confusion_matrix, f1_score, precision_score, recall_score, classification_report
import seaborn as sns
from scipy.sparse import csr_matrix
```

Load dữ liệu

Load dữ liệu

```
In [2]: df = pd.read_csv("https://media.githubusercontent.com/media/PTIT-Assignment-Projects/ai-svm-email-spam/refs/heads/main/dataset/sampled_dataset2000.csv")
```

Import thư viện

```
In [3]: df.head()
```

Out[3]:

	label	text
0	1	wireless optical\n3 - button scroll mouse\nlim...
1	1	from the desk of philip moore\ndept credit con...
2	1	submitting your website in search engines may ...
3	1	managers wanted inc escapenumber company is lo...
4	1	anatrim an extremely efficient flesh loss blen...

Xoá những ký tự đặc biệt và số

In [10]:

```
def remove_special_characters(word):  
    return re.sub(r'[^a-zA-Z\s]', '', word)
```

In [11]:

```
text = 'Hello everyone ! I am happy to meet 3 of us, my email is admin@gmail.com'
print(remove_special_characters(text))
```

Hello everyone I am happy to meet of us my email is admingmailcom

Xoá những stop words trong câu

```
In [12]: ENGLISH_STOP_WORDS = set(stopwords.words('english'))
```

```
In [13]: def remove_stop_words(words):  
         return [word for word in words if word not in ENGLISH_STOP_WORDS]
```

```
In [14]: print(' '.join(remove_stop_words(text.split())))
```

Hello everyone ! I happy meet 3 us, email admin@gmail.com

Xoá các link website trong câu

Xoá các link website trong câu

In [15]:

```
def remove_url(word):
    return re.sub(r"http\S+", "", word)
```

In [16]:

```
text = 'My websites are https://google.com and https://reddit.com'
print(remove_url(text))
```

My websites are and

Áp dụng word_tokenize vào data

In [17]:

```
df['text'] = df['text'].apply(remove_special_characters)
df['text'] = df['text'].apply(remove_url)
df['text'] = df['text'].apply(word_tokenize)
df['text'] = df['text'].apply(remove_stop_words)
df['text'] = df['text'].apply(' '.join)
```

In [18]:

```
df.head()
```

Out[18]:

	label	text
0	1	wireless optical button scroll mouse limited s...
1	1	desk philip moore dept credit control unit uni...
2	1	submitting website search engines may increase...
3	1	managers wanted inc escapenumber company looki...
4	1	anatrim extremely efficient flesh loss blend r...

Snowball Stemmer

Word	Stem
cared	care
university	univers
fairly	fair
easily	easili
singing	sing
sings	sing
sung	sung
singer	singer
sportingly	sport

Snowball Stemmer

In [19]:

```
stemmer = SnowballStemmer('english')

def stem_text(text):
    tokens = nltk.word_tokenize(text)

    stemmed_tokens = [stemmer.stem(token) for token in tokens]

    return ' '.join(stemmed_tokens)
```

In [20]:

```
df['text'] = df['text'].apply(stem_text)
df.head()
```

Out[20]:

	label	text
0	1	wireless optic button scroll mous limit stock ...
1	1	desk philip moor dept credit control unit unio...
2	1	submit websit search engin may increas onlin s...
3	1	manag want inc escapenumb compani look manag g...
4	1	anatrim extrem effici flesh loss blend readili...

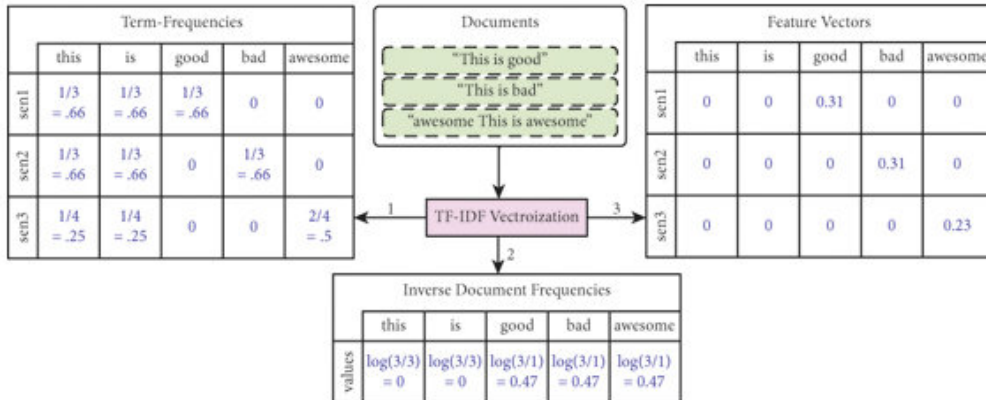
TF-IDF vectorization

$$TF = \frac{\text{Number of times a word "X" appears in a Document}}{\text{Number of words present in a Document}}$$

$$IDF = \log \left(\frac{\text{Number of Documents present in a Corpus}}{\text{Number of Documents where word "X" has appeared}} \right)$$

$$TF\ IDF = TF * IDF$$

TF-IDF vectorization



TF-IDF vectorization

TF-IDF vectorization

<https://www.geeksforgeeks.org/understanding-tf-idf-term-frequency-inverse-document-frequency/>

```
[21]: vectorizer = TfidfVectorizer()

X = vectorizer.fit_transform(df['text'])
print(X.shape)
```

```
(83448, 234926)
```


Chuẩn bị dữ liệu train và test

Chuẩn bị dữ liệu train và test

In [22]:

```
y = df['label']
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```


HardMarginSVM

Kết quả

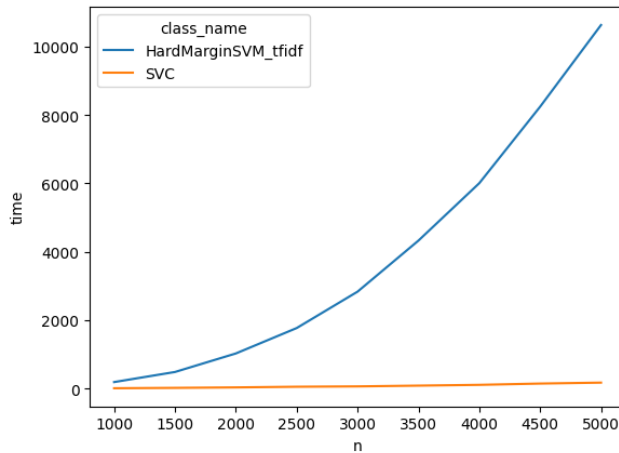
Đang huấn luyện SVM...
Thời gian chạy: 1030.955427646637

```
=== Đánh giá Hard Margin SVM ===
===== Kết quả đánh giá Hard Margin SVM =====
Accuracy: 0.9450
Precision: 0.9375
Recall: 0.9559
F1-score: 0.9466
```

Báo cáo chi tiết:

	precision	recall	f1-score	support
0	0.95	0.93	0.94	196
1	0.94	0.96	0.95	204
accuracy			0.94	400
macro avg	0.95	0.94	0.94	400
weighted avg	0.95	0.94	0.94	400

So sánh thời gian chạy với độ lớn bộ dữ liệu



1 Giới thiệu

② Lý thuyết

③ Cài đặt

Tiền xử lý và chuẩn bị dữ liệu

Mô hình HardMarginSVM

Mô hình SVM tối ưu bởi thuật toán Pegasos

Viết mô hình

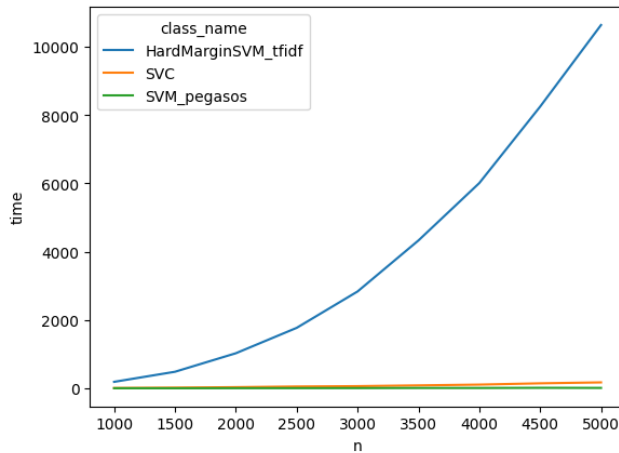
So sánh với HardMarginSVM

Chạy với bộ dữ liệu ban đầu và lưu mô hình

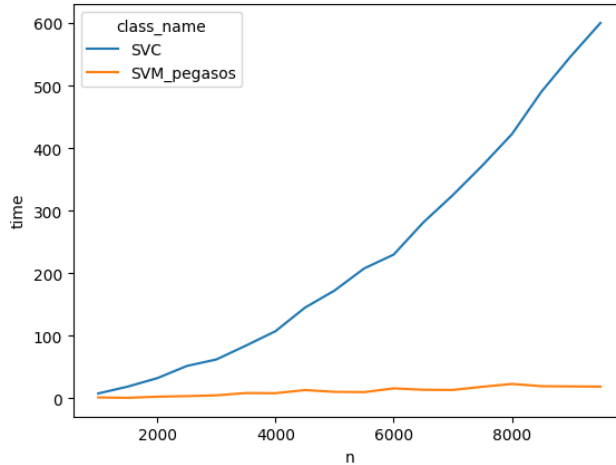
④ Triển khai và demo

71

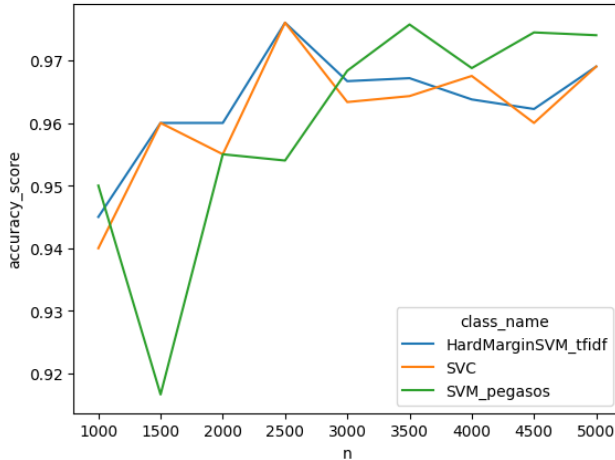
So sánh thời gian chạy với độ lớn bộ dữ liệu



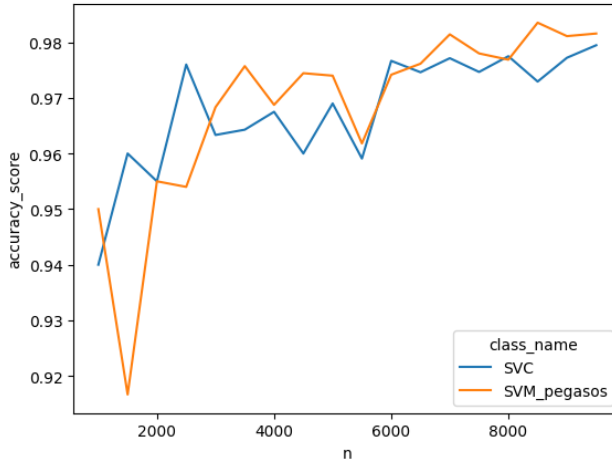
So sánh thời gian chạy với độ lớn bộ dữ liệu



So sánh độ lớn bộ dữ liệu với độ chính xác



So sánh độ lớn bộ dữ liệu với độ chính xác



Kết quả

Start preprocessing data

(66758, 206343)

End preprocessing data

Epoch 10, Giá trị hàm mục tiêu: 0.1199

Epoch 20, Giá trị hàm mục tiêu: 0.1176

Dừng sớm tại epoch 22, giá trị hàm mục tiêu thay đổi: 0.000032

	class_name	n	time	prep_time	accuracy_score	f1_score
0	SVM	83448	261.033806	283.776974	0.981186	0.982282

Lưu mô hình và vectorizer

```
model_filename = f'linear_svm.pkl'
vectorizer_filename = f'vectorizer.pkl'

with open(model_filename, 'wb') as model_file:
    pickle.dump(svm_base, model_file)

with open(vectorizer_filename, 'wb') as vectorizer_file:
    pickle.dump(vectorizer, vectorizer_file)
```

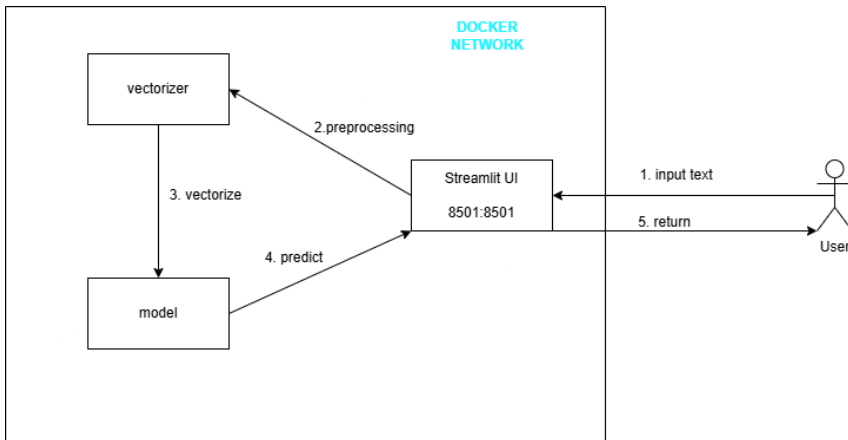
- 1 Giới thiệu
 - 2 Lý thuyết
 - 3 Cài đặt
 - 4 Triển khai và demo
- Mô tả cách triển khai

- 1 Giới thiệu
- 2 Lý thuyết
- 3 Cài đặt
- 4 Triển khai và demo
Mô tả cách triển khai

Mô tả cách triển khai

- Sử dụng streamlit để tạo giao diện trên web và xử lý phần mô hình
- Load 2 file nhị phân model và vectorizer để tiến hành dự đoán đầu vào
- Sử dụng docker để đóng gói

Sơ đồ tổng quan



Demo phân loại email tiếng Anh spam

Nhập nội dung email:

Add Information, Not Clutter with Shape Data
Need to add more information to your diagram, but don't want to end up with gigantic shapes with huge blocks of text? Shape data lets you enrich a shape with information and it's viewable as a tooltip. You can add cost, description, asset number, manufacturer, and so much more. Add information like asset numbers and descriptions without clutter

Kiểm tra

 Email có khả năng là SPAM

Văn bản sau khi được tiền xử lý:

add inform not clutter shape data need add inform diagram dont want end gigant shape huge block text
shape data let enrich shape inform viewabl tooltip you add cost descript asset number manufactur much
add inform like asset number descript without clutter

Demo phân loại email tiếng Anh spam

Nhập nội dung email:

The report suggests that universities will need to look beyond their traditional international student recruitment markets, given that demand in India is slowing and higher education is improving in quality in east Asia. They may also need to consider offering more cost-effective options, the report notes.

Kiểm tra

☒ Email có khả năng không phải là SPAM

Văn bản sau khi được tiền xử lý:

the report suggest univers need look beyond tradit intern student recruit market given demand india slow
higher educ improv qualiti east asia they may also need consid offer costeffect option report note

Cảm ơn cô đã lắng nghe!